International Journal of Science Engineering and Advance Technology,IJSEAT, Vol 3, Issue 4,APRIL - 2015 | ISSN 2321-6905

International Journal of Science Engineering and Advance Technology

Combination Methodology For Protected Formal Deduplication In Cloud Computing

Joykumar.S , Chandra Bhushana Rao.K
#Silaparapu Joykumar M.Tech (Phd), Asst.Prof Department of Computer Science and Engineering, GDMM Engineering And Technology, Nandigama
Killi. Chandra Bhushana Rao M.Tech (Phd), Asst.Prof Department of Computer Science and Engineering, GDMM Engineering And Technology, Nandigama

Abstract:
Data Deduplication is an significant method in favour get rid of redundant data as an alternative of captivating files, it provisions merely distinct copy of file. Among the entire organizations many organizations, storage schema enclose more pieces of duplicate data. . For example, Different users stores similar files in several different places. Deduplication abolish these additional copies by saving as single copy of data and reinstate the other copies along with pointers that flipside to the original copy. It be the data compression technique for to increase the bandwidth efficiency and storage exploitation. Data Deduplication is enormously using in cloud computing now a days. It formulate data management scalable and storage setback in cloud computing. Data Deduplication defend the confidentiality of sensitive information. It works with convergent encryption technique to encrypt the data before uploading. . Companies regularly use Deduplication in backup and calamity recovery applications. In this paper we attempt the authorized Deduplication check, merge with convergent encryption to afford security for sensitive data with hybrid cloud computing.

Key Words: Deduplication, authorized duplicate check, confidentiality.

I. Introduction:

Cloud computing technique is extensively used now a days. In that, computing is used in the large communication network in the vein of Internet. It is the significant solution for business storage in very low cost. Cloud computing afford huge amount of storage in all sectors like government. In this enterprise also used for storing our secret data on cloud, Without any backdrop implementation details. All platforms users can access and shares dissimilar resources on cloud. The most significant problem in cloud computing is bulk amount of storage space and security concerns. One critical challenge of cloud storage to managing of forever-increasing volume of data. To enhance scalability, storage problem data Deduplication is the most imperative technique and concerned more attention now a days. It is the significant technique for data compression , It evade the duplicate copies of data and store one copy of data. Data Deduplication takes place in neither block level nor file level. In file level approach duplicate files get rid of, and in block level approach duplicate blocks of data takes place in non-identical files. Deduplication diminish the storage needs equipped to 90-95% for backup application and 68% in standard file system. The vital problem in data Deduplication is security and privacy for to protect the data from insider or outsider attack. For data confidentiality and encryption, different user actually encrypt there data or files. For this user using a secrete key to perform encryption and decryption operation. For uploading a file to an cloud user he first generate convergent key and encryption of the file then finally loads the file to the cloud. To prevent unauthorized access control the ownership protocol is used to provide proof that the user indeed owns the same file when Deduplication occurs. After the proof, server afford a pointer to subsequent user for accessing same file without needing to upload same file. When user wants to download those files, he will download encrypted file from cloud and next decrypt's that file using convergent key.

II. Preliminaries

In this section, we first define the notations used in this paper, review some secure primitives used in our secure Deduplication.

Symmetric encryption. Symmetric encryption uses a common secret key *to* encrypt and decrypt information.

A symmetric encryption scheme consists of three Primitive functions:

- KeyGenSE(1_) *!* κ is the key generation algorithm that generates κ using security parameter 1;
- DecSE(κ,*C*) *!* *M* is the symmetric decryption algorithm that takes the secret κ and ciphertext *C* and

then outputs the original message *M*.

- EncSE(κ,*M*) *!* *C* is the symmetric encryption algorithm that takes the secret κ and message *M* and

then outputs the ciphertext *C*.

Convergent encryption. Convergent encryption [3], [4] provides data confidentiality in Deduplication. A data owner derives a
convergent key from each original data copy and encrypts the data copy with the convergent key.

- KeyGenCE(*M*) *! K* is the key generation algorithm that maps a data copy *M* to a convergent key *K*
- EncCE(*K*,*M*) *! C* is the symmetric encryption algorithm that takes both the convergent key *K* and the data copy *M* as
inputs and then outputs a ciphertext *C*.
- DecCE(*K*,*C*) *! M* is the decryption algorithm that takes both the ciphertext *C* and the convergent key *K* as inputs and then outputs the original data copy *M*; and
- TagGen(*M*) *! T*(*M*) is the tag generation algorithm that maps the original data copy *M* and outputs a tag *T*(*M*).
- DecSE(*κ*,*C*) *! M* is the symmetric decryption algorithm that takes the secret *κ* and ciphertext *C* and
then outputs the original message *M*.

Proof of ownership. The notion of proof of ownership (PoW) [2] enables users to prove their ownership of
data copies to the storage server. However PoW is implemented as an interactive algorithm (denoted by
PoW) run by a prover (i.e., user) and a verifier (i.e.storage server).

Identification Protocol. An identification protocol can be described with two phases: Proof and Verify. In the stage of Proof,
a prover/user *U* can demonstrate his identity to a verifier by performing some identification proof related to his identity. The
input of the prover/user is his private key *skU* that is sensitive information such as private key of a public key in his certificate
or credit card number etc. that he would not like to share with the other users.

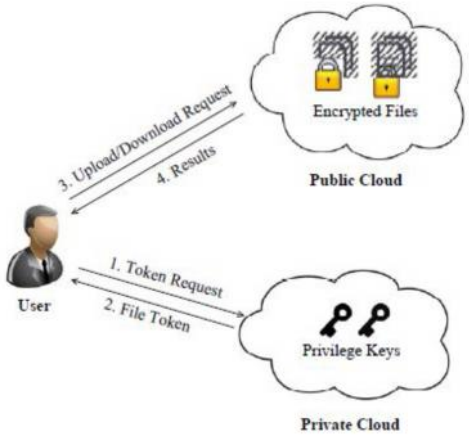

Fig. 1. Architecture for Authorized Deduplication

III. System Model:

Hybrid Architecture for Secure Deduplication

At a high level, our setting of interest is an enterprise network, consisting of a group of affiliated clients (for example, employees of a company) who will use the S-CSP and store data with Deduplication technique. The S-CSP performs Deduplication by checking if the contents of two files are the same and stores only one of them. Each privilege is represented in the form of a short message called *token*.

- S-CSP. This is an entity that provides a data storage service in public cloud. The S-CSP provides the data outsourcing service and stores data on behalf of the users.
- Data Users. A user is an entity that wants to outsource data storage to the S-CSP and access the data later. In a storage system supporting Deduplication, the user only uploads unique data but does not upload any duplicate data to save the upload bandwidth, which may be owned by the same use or different users. Every single file is protected with the convergent encryption key and privilege keys to realize the authorized Deduplication with differential privileges.
- Private Cloud. Compared with the traditional Deduplication architecture in cloud computing, this is a new entity introduced for facilitating user's secure usage of cloud service. Private Keys are managed by private cloud in order to give them privileges as per their designation.

IV. DESIGN GOALS :

In this paper, we address the problem of privacy preserving Deduplication in cloud computing and propose a new Deduplication system supporting for:

- **Differential Authorization**. Each authorized user is able to access its individual token of his file to perform duplicate check based on authority.Under this assumption, any user cannot generate a token for duplicate check out of his access or without the aid from the private cloud server.
- **Authorized Duplicate Check.** Authorized user is able to access his/her own token from private cloud, while the public cloud performs duplicate check directly and tells the user if there is any duplicate. The security requirements considered in this paper lie in two folds, including the security of file token and security of data files. For the security of file token, two aspects are defined as unforgeability and indistinguishability of file token. The details are given below.
- **Unforgeability of file token/duplicate-check token.** User make registration in private cloud for generating file token.Using respective file token he/she upload or download files on public cloud. The users are not allowed to collude with the public cloud server to break the unforgeability of file

tokens. In our system, the S-CSP is honest but curious and will honestly perform the duplicate check upon receiving the duplicate request from users. The duplicate check token of users should be issued from the private cloud server in our scheme.

Indistinguishability of file token/duplicate-check token.

It requires that any user without querying the private cloud server for some file token, he cannot get any useful information from the token, which includes the file information and key information.

• **Data Confidentiality**. Unauthorized users without appropriate token , including the S-CSP and the private cloud server, should be prevented from access to the underlying plaintext stored at S-CSP. In another word, the goal of the adversary is to retrieve and recover the files that do not belong to them. In our system, compared to the previous definition of data confidentiality based on convergent encryption, a higher level confidentiality is defined and achieved.

V. Security Analysis:

Proposed system has been designed to solve the differential privilege problem in secure Deduplication. The security will be analyzed in terms of two aspects, that is, the authorization of duplicate check and the confidentiality of data.

Security of Duplicate-Check Token

We consider several types of privacy we need protect, that is, i) unforgeability of duplicate-check token: There are two types of adversaries, that is, external adversary and internal adversary. As shown below, the external adversary can be viewed as an internal adversary without any privilege. If a user has privilege *p*, it requires that the adversary cannot forge and output a valid duplicate token with any other privilege *p′* on any file *F*, where *p* does not match *p′*. Furthermore, it also requires that if the adversary does not make a request of token with its own privilege from private cloud server, it cannot forget and output a valid duplicate token with *p* on any *F* that has been queried. The internal adversaries have more attack power than the external adversaries and thus we only need to consider the security against the internal attacker, ii) indistinguishability of duplicate check token : this property is also defined in terms of two aspects as the definition of unforgeability. First, if a user has privilege *p*, given a token ϕ', it requires that the adversary cannot distinguish which privilege or file in the token if *p* does not match *p′*.

VI. Proposed System:

In our system we implement a project that includes the public cloud and the private cloud and also the hybrid cloud which is a combination of the both public cloud and private cloud. In general by if we used the public cloud we can't provide the security to our private data and hence our private data will be loss.So that we have to provide the security to our data for that we make a use of private cloud also. When we use a private clouds the greater security can be provided.In this system we also provides the data Deduplication . which is used to avoid the duplicate copies of data.User can upload and download the files from public cloud but private cloudprovides the security for that data.that means only the authorized person can upload and download the files from the

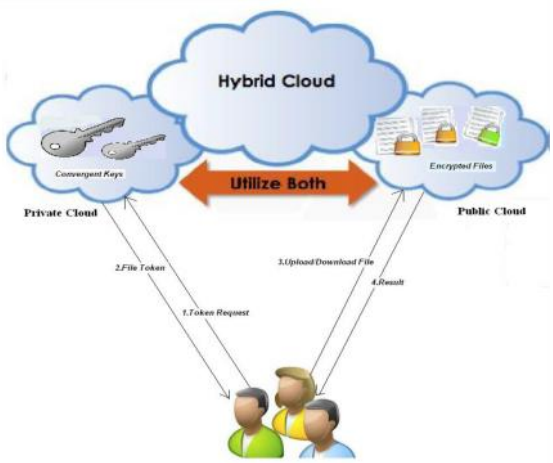

Fig2: Architecture of Authorized Deduplication

public cloud.for that user generates the key and storeed that key onto the private cloud. at the time of downloading user request to the private cloud for key and then access that Particular file.

VII. Implementation:

We implement a prototype of the proposed authorized Deduplication system, in which we model three entities as separate C++ programs. A *Client* program is used to model the data users to carry out the file upload process. A *Private Server* program is used to model the private cloud which manages the private keys and handles the file token computation. A *Storage Server* program is used to model the S-CSP which stores and deduplicate files.

Our implementation of the **Client** provides the following function calls to support token generation and Deduplication along the file upload process.

• FileTag(File) - It computes SHA-1 hash of the File as File Tag;

• TokenReq(Tag, UserID) - It requests the Private Server for File Token generation with the File Tag and User ID;
• DupCheckReq(Token) - It requests the Storage Server for Duplicate Check of the File by sending the file token received
from private server;
• ShareTokenReq(Tag, {Priv.}) - It requests the Private Server to generate the Share File Token with the File Tag and
Target Sharing Privilege Set;
• FileEncrypt(File) - It encrypts the File with Convergent Encryption using 256-bit AES algorithm in cipher block chaining
(CBC) mode, where the convergent key is from SHA-256 Hashing of the file;
• FileUploadReq(FileID, File, Token) – It uploads the File Data to the Storage Server if the file is Unique and updates the
File Token stored. Our implementation of the **Private Server** includes corresponding request handlers for the token
generation and maintains a key storage with Hash Map.
• TokenGen(Tag, UserID) - It loads the associated privilege keys of the user and generate the token with HMAC-SHA-1
Algorithm

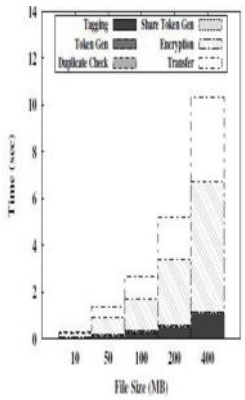

Fig. 2. Time Breakdown for Different File Size

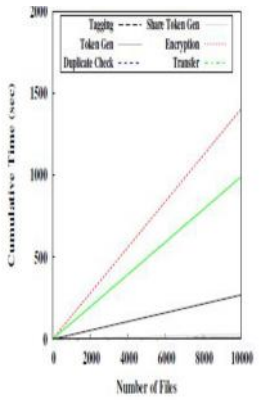

Fig. 3. Time Breakdown for Different Number of Stored Files

VIII. Assessment:
Our evaluation focuses on comparing the overhead induced by authorization steps, including file token generation and share
token generation, against the convergent encryption and file upload steps. We evaluate the overhead by varying different
factors, including 1) File Size 2) Number of Stored Files 3) Deduplication Ratio 4) Privilege Set Size. We break down the
upload process into 6 steps, 1) Tagging 2) Token Generation 3) Duplicate Check 4) Share Token Generation 5) Encryption 6)
Transfer . For each step, we record the start and end time of it and therefore obtain the breakdown of the total time spent. We
present the average time taken in each data set in the figures

File Size
To evaluate the effect of file size to the time spent on different steps, we upload 100 unique files (i.e., without any
Deduplication opportunity) of particular file size and record the time break down. Using the unique files enables us to evaluate
the worst-case scenario where we have to upload all file data. The average time of the steps from test sets of different file size
are plotted in Figure 2. The time spent on tagging, encryption, upload increases linearly with the file size, since these
operations involve the actual file data and incur file I/O with the whole file.

Number of Stored Files
To evaluate the effect of number of stored files in the system, we upload 10000 10MB unique files to the system and record the
breakdown for every file upload. From Figure 3, every step remains constant along the time. Token checking is done with a hash table and a linear search would be carried out in case of collision.

Deduplication Ratio
To evaluate the effect of the Deduplication ratio, we prepare two unique data sets, each of which consists of 50 100MB files.
We first upload the first set as an initial upload. For the second upload, we pick a portion of 50 files, according to the given
Deduplication ratio, from the initial set as duplicate files and remaining files from the second set as unique files. The average
time of uploading the second set is presented in Figure.

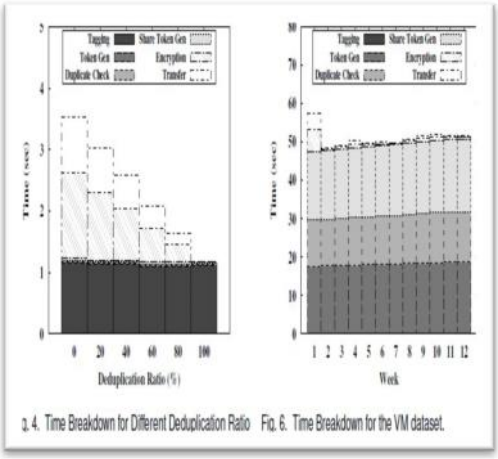

Fig. 4. Time Breakdown for Different Deduplication Ratio Fig. 6. Time Breakdown for the VM dataset.

IX. Conclusion:

In this paper, the idea of authorized data Deduplication was proposed to protect the data security by including differential authority of users in the duplicate check. In public cloud our data are securely store in encrypted format,and also in private cloud our key is store with respective file.There is no need to user remember the key.So without key anyone can not access our file or data from public cloud. As a proof of concept, we implemented a prototype of our proposed authorized duplicate check scheme and conduct testbed experiments on our prototype. We showed that our authorized duplicate check scheme incurs minimal overhead compared to convergent encryption and network transfer.

X. References:

[1] OpenSSL Project. http://www.openssl.org/.
[2] P. Anderson and L. Zhang. Fast and secure laptop backups with
encrypted de-duplication. In *Proc. of USENIX LISA*, 2010.
[3] M. Bellare, S. Keelveedhi, and T. Ristenpart. Dupless: Serveraided
encryption for deduplicated storage. In *USENIX Security*
Symposium, 2013.
[4] M. Bellare, S. Keelveedhi, and T. Ristenpart. Message-locked
encryption and secure Deduplication. In *EUROCRYPT*, pages 296–
312, 2013.
[5] M. Bellare, C. Namprempre, and G. Neven. Security proofs for
identity-based identification and signature schemes. *J. Cryptology*,
22(1):1–61, 2009.
[6] M. Bellare and A. Palacio. Gq and schnorr identification schemes:
Proofs of security against impersonation under active and concurrent
attacks. In *CRYPTO*, pages 162–177, 2002.
[7] S. Bugiel, S. Nurnberger, A. Sadeghi, and T. Schneider. Twin
clouds: An architecture for secure cloud computing. In *Workshop*
on Cryptography and Security in Clouds (WCSC 2011), 2011.
[8] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer.
Reclaiming space from duplicate files in a serverless distributed
file system. In *ICDCS*, pages 617–624, 2002.
[9] D. Ferraiolo and R. Kuhn. Role-based access controls. In *15th*
NIST-NCSC National Computer Security Conf., 1992.
[10]GNULibmicrohttpd.
http://www.gnu.org/software/libmicrohttpd/.
[11] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg. Proofs of
ownership in remote storage systems. In Y. Chen, G. Danezis,
and V. Shmatikov, editors, *ACM Conference on Computer and*
Communications Security, pages 491–500. ACM, 2011.

Authors:

Silaparapu Joykumar M.Tech (Phd), Asst.Prof Department of Computer Science and Engineering, GDMM Engineering And Technology, Nandigama

Killi. Chandra Bhushana Rao M.Tech (Phd), Asst.Prof Department of Computer Science and Engineering, GDMM Engineering And Technology, Nandigama